LES BONNES PRATIQUES DU DEVOPS

Introduction

Le DevOps vient avec ces bonnes pratiques et fournit certainement un support étendu pour mener à bien le projet en établissant une collaboration et une communication de niveau supérieur entre les entreprises et les fournisseurs de services logiciels.

Pour une collaboration efficace entre les équipes de développement et d'exploitation, le DevOps fournit une variété de bonnes pratiques, que nous allons découvrir dans cette partie.

Les bonnes pratiques

Participation active des parties prenantes

Un aspect fondamental de la philosophie DevOps est que les développeurs et le personnel d'exploitation doivent **travailler en étroite collaboration régulièrement**. Cela implique qu'ils doivent se considérer mutuellement comme des parties prenantes importantes et chercher activement à travailler ensemble.

Suivi et de contrôle de version

L'**implémentation d'outils de suivi et de contrôle de version** peut aider l'entreprise à obtenir une meilleure visibilité sur la progression d'un projet et à collaborer facilement avec des équipes réparties.

Un système de contrôle de versions pour les différentes équipes tel que Git, permet de tracer les modifications dans la base de code d'une application ou d'une configuration et il devient plus simple quand le besoin se manifeste, de faire un retour en arrière. Ainsi, en permettant aux équipes de collaborer et d'intégrer les modifications dans le référentiel* partagé, cela améliore considérablement le cycle de vie de l'application.

> **Définition**
>
> ***Référentiel** : nommé également « dépôt » ou « Repository » en anglais, c'est un stockage centralisé et organisé de données.

Les microservices

L'architecture des microservices est une approche de conception pour **construire une seule application comme un ensemble de petits services**. Chaque service s'exécute dans son propre processus et communique avec d'autres services via une interface bien définie à l'aide d'un mécanisme léger, généralement une API* (Application Programming Interface) basée sur le protocole HTTP(S). Cette architecture présente de nombreux avantages pour les équipes usant d'un modèle DevOps.

> **Définition**
>
> ***API** : solution informatique qui offre la possibilité à plusieurs applications de communiquer entre elles et d'échanger des données.

Inversement, une application monolithique est conçue comme une seule unité autonome. Une modification apportée à une petite section de code affecte l'ensemble du système et peut nécessiter la création et le déploiement d'une toute nouvelle version du logiciel. Concernant la disponibilité d'une application avec une architecture monolithique, sa mise à l'échelle signifie également qu'il est nécessaire de mettre à l'échelle l'ensemble de l'application.

Les microservices résolvent les problèmes des architectures monolithiques en étant aussi modulaires que possible. En effet, ils aident à créer une application en tant que suite de petits services, chacun s'exécutant dans son propre processus et sont déployables indépendamment. Ces services peuvent être codés dans des langages de programmation distincts et peuvent utiliser différentes technologies de stockage de données.

Enfin, les microservices reposent sur des capacités commerciales, chaque service est limité à un seul objectif et peut être déployé indépendamment, en tant que service unique ou en tant que groupe de services.

Automatisation des tests

Afin de livrer un code de qualité, les développeurs doivent **tester régulièrement leurs applications**. Ici, le DevOps recommande d'automatiser ces tests afin de permette aux développeurs d'identifier et de résoudre plus rapidement et prématurément les erreurs pendant le développement du logiciel plutôt que plus tard dans un environnement de production.

L'automatisation des tests peut être déclenchée aux modifications de la base de données et du réseau, aux configurations de middleware* et au développement de code à l'aide de tests unitaires.

Définition

***Middleware** : logiciel tiers qui permet un échange d'informations entre différentes applications.

Gestion du changement

La gestion du changement est un processus dans lequel les configurations sont modifiées et redéfinies pour répondre aux conditions de circonstances dynamiques et aux nouvelles exigences.

Du point de vue informatique, cette gestion est l'acte d'**assurer une évolution réussie de l'infrastructure** pour mieux soutenir l'organisation globale.

Cette gestion peut être assez délicate pour l'équipe du projet car de nombreuses technologies, et même différentes versions de technologies similaires, seront utilisées dans le développement d'une solution. En raison de la nécessité de considérer un grand nombre de solutions fonctionnant et interagissant en production simultanément, cela rend plus difficile ce management.

Pour rendre efficace une gestion intégrée du changement dans une entreprise, les équipes de développement doivent travailler en étroite collaboration avec les équipes opérationnelles pour **comprendre les implications de tout changement technologique au niveau de l'organisation**. Les équipes opérationnelles doivent également fournir leurs contributions sur les opportunités et les conséquences que le changement pourrait exposer à un niveau plus large et sur les autres systèmes susceptibles d'être impactés.

Intégration continue

Dans un modèle DevOps, la CI (intégration continue) s'agit d'une pratique de développement de logiciels DevOps où l'équipe de développement met régulièrement à jour les modifications de code dans un logiciel de contrôle de versions, après quoi les phases de builds et de tests s'exécutent automatiquement. Les pratiques d'intégration continue permettent aux développeurs d'effectuer des intégrations plus tôt et fréquemment.

Les objectifs clés de l'intégration continue sont de trouver et de **résoudre les bogues plus rapidement**, d'améliorer la qualité des logiciels et de réduire le temps nécessaire pour valider et publier les nouvelles mises à jour logicielles. Ainsi, la CI améliore les collaborations entre les équipes et construit finalement un produit logiciel de haute qualité.

Livraison/Déploiement continue

La livraison continue est une pratique DevOps qui se développe sur l'intégration continue depuis le code nouvellement développé, mis à jour, testé, et buildé automatiquement par les développeurs.

Une fois que ces tests réussissent toutes les épreuves, il est automatiquement préparé pour une MEP (Mise En Production). Et lorsqu'ils sont correctement mis en œuvre, les exploitants auront toujours un artefact* de build prêt pour le déploiement qui a passé au préalable par un processus de test standardisé.

> **Définition**
>
> ***artefact** : produit, qui résulte du processus d'un développement logiciel.

La différence entre la livraison continue et le déploiement continu est la présence d'une approbation manuelle pour la mise à jour en production. Avec un déploiement continu, l'application passe en production automatiquement et sans approbation explicite.

Ces pratiques aident les organisations à **réduire les travaux manuels et à augmenter la fréquence et la vitesse de livraison**.

Surveillance des applications

Il s'agit de la pratique opérationnelle de monitorer l'infrastructure et les applications en cours d'exécution dans un environnement de production. Les systèmes d'exploitation, les serveurs d'applications, les services de communication et les applications offrent souvent des métriques de surveillance qui peuvent être exploitées par des outils de surveillance.

La surveillance de l'infrastructure des applications est très cruciale pour **optimiser les performances des applications**. Il est donc très important pour les équipes de développement et les équipes d'exploitation d'**envisager une surveillance proactive** et de vérifier les performances de leurs applications.

L'infrastructure en tant que code

Auparavant, la gestion des infrastructures informatique était un travail laborieux. Les administrateurs systèmes devaient aménager et configurer manuellement tout le matériel et les logiciels nécessaires au fonctionnement des applications. Avec l'IaC (Infrastructure as Code), cette gestion est automatisée.

> **Information**
>
> J'ai dédié un cours complet sur Terraform qui est un outil d'IaC très utilisé par les entreprises.

En effet, l'IaC utilise un langage de codage descriptif de haut niveau pour **automatiser l'approvisionnement de l'infrastructure informatique**. Cette automatisation élimine la nécessité pour les équipes de provisionner et de gérer manuellement les serveurs, les systèmes d'exploitation, les connexions aux bases de données, le stockage et d'autres éléments d'infrastructure chaque fois qu'ils souhaitent développer, tester ou déployer une application logicielle.

Cette tendance, permet à une organisation de plus facilement contrôler ses coûts, diminuer ses risques et répliquer rapidement aux nouvelles opportunités commerciales et aux menaces concurrentielles.

De plus, de la même manière que les développeurs suivent les modifications de leurs applications à travers un outil de versioning, les équipes systèmes peuvent effectuer la même chose avec l'IaC, en suivant et traçant l'évolution de leur infrastructure depuis un outil de versioning. De ce fait, en cas de besoin d'un retour en arrière sur une ancienne infrastructure, il suffira juste de récupérer l'ancienne version du code de l'infrastructure et de l'exécuter.

Une question de culture

Après avoir décrit ces pratiques critiques qui prennent en charge le DevOps, je ressens le besoin de souligner que le principal facteur de la réussite d'une implémentation du DevOps, c'est de **créer une culture collaborative et respectueuse**

dans l'ensemble d’une organisation informatique.

En effet, d'après mon expérience, les gens et la façon dont ils travaillent ensemble reste le principal déterminant du succès lorsqu'il s'agit d'adopter une stratégie DevOps efficace. Cela permet aux équipes d’établir des règles culturelles fortes autour du partage d'informations, et de s'aligner plus étroitement sur un objectif clair et commun.

Conclusion

Le DevOps est une stratégie que la majorité des sociétés développant des logiciels aspirent à fournir à leurs clients dans le but de fournir rapidement des applications de haute qualité en établissant la transparence et une collaboration ouverte entre leurs équipes de développement et leurs équipes opérationnelles.

En suivant les pratiques DevOps mentionnées ci-dessus, les fournisseurs de services informatiques peuvent s'assurer à fournir des solutions logicielles qui aideront les entreprises à parvenir inéluctablement à leur objectif métier.